Formerly DA-IICT

IT643: Software Design and Testing

Faculty Name: Prof. Ankush Chander

Project Name: GreenCart

Group Name: NexTech

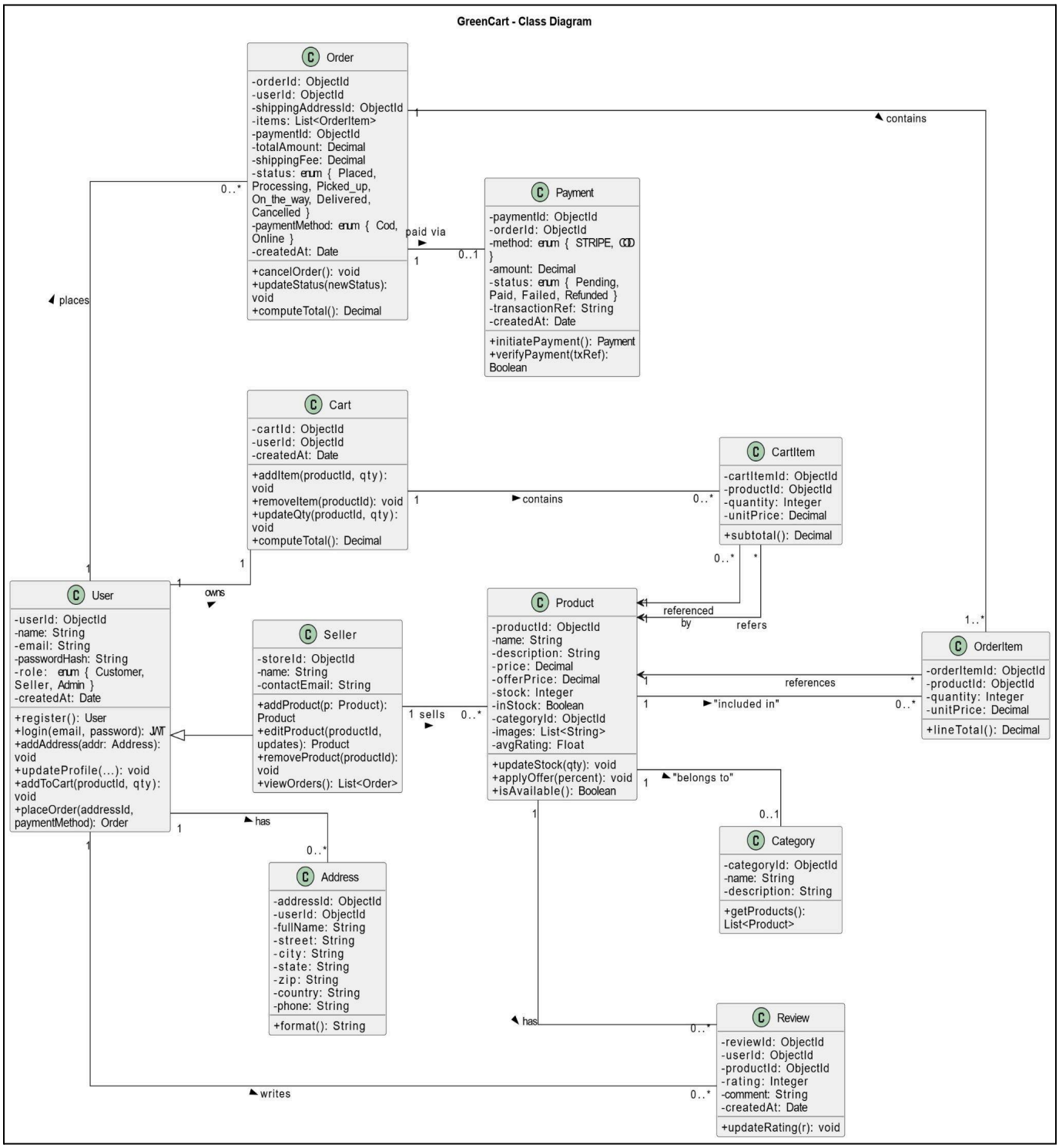
Team Members

202412101 : Shah Yashvi Devangkumar

202412119 : Ritik Kalal

202412109 : Stuti M. Shah

GreenCart Class Diagram



GreenCart - Class Diagram Documentation

1. Introduction:-

The Class Diagram of the GreenCart application represents the static structure of the system by illustrating its major classes, attributes, operations, and the relationships among them. This diagram provides a clear overview of how various components interact within the application, including user management, product catalog, shopping cart, orders, payments, and reviews. The objective is to visualize system behavior and support effective development, maintenance, and scalability.

2. Classes And Their Descriptions:-

2.1 User:-

Attributes:

- userId: ObjectId
- name: String
- email: String
- passwordHash: String
- role: enum { CUSTOMER, SELLER, ADMIN }
- createdAt: Date

Methods:

- register(): User
- login(email, password): JWT
- addAddress(addr: Address): void
- updateProfile(...): void
- addToCart(productId, qty): void
- placeOrder(addressId, paymentMethod): Order

2.2 Seller:-

(Represents a User with selling privileges)

Attributes:

- storeId: ObjectId
- name: String
- contactEmail: String

Methods:

- addProduct(p: Product): Product
- editProduct(productId, updates): Product
- removeProduct(productId): void
- viewOrders(): List<Order>

2.3 Product:-

Attributes:

- productId: ObjectId
- name: String
- description: String
- price: Decimal
- offerPrice: Decimal
- stock: Integer
- inStock: Boolean
- categoryId: ObjectId
- images: List<String>
- avgRating: Float

Methods:

- updateStock(qty): void
- applyOffer(percent): void
- isAvailable(): Boolean

2.4 Category:-

Attributes:

- categoryId: ObjectId
- name: String
- description: String

Methods:

- getProducts(): List<Product>

2.5 Cart:-

Attributes:

- cartId: ObjectId
- userId: ObjectId
- createdAt: Date

Methods:

- addItem(productId, qty): void
- removeItem(productId): void
- updateQty(productId, qty): void
- computeTotal(): Decimal

2.6 CartItem:-

Attributes:

- cartItemId: ObjectId
- productId: ObjectId
- quantity: Integer
- unitPrice: Decimal

Methods:

- subtotal(): Decimal

2.7 Order:-

Attributes:

- orderId: ObjectId
- userId: ObjectId
- shippingAddressId: ObjectId
- items: List<OrderItem>
- paymentId: ObjectId
- totalAmount: Decimal
- shippingFee: Decimal

- status: enum { PLACED, PROCESSING, PICKED_UP, ON_THE_WAY, DELIVERED, CANCELLED }
- paymentMethod: enum { COD, ONLINE }
- createdAt: Date

Methods:

- cancelOrder(): void
- updateStatus(newStatus): void
- computeTotal(): Decimal

2.8 OrderItem:-

Attributes:

- orderItemId: ObjectId
- productId: ObjectId
- quantity: Integer
- unitPrice: Decimal

Methods:

- lineTotal(): Decimal

2.9 Address:-

Attributes:

- addressId: ObjectId
- userId: ObjectId
- fullName: String
- street: String
- city: String
- state: String
- zip: String
- country: String
- phone: String

Methods:

- format(): String

2.10 Payment:-

Attributes:

- paymentId: ObjectId
- orderId: ObjectId
- method: enum { STRIPE, COD }

- amount: Decimal
- status: enum { PENDING, PAID, FAILED, REFUNDED }
- transactionRef: String
- createdAt: Date

Methods:

- initiatePayment(): Payment
- verifyPayment(txRef): Boolean

2.11 Review:-

Attributes:

- reviewId: ObjectId
- userId: ObjectId
- productId: ObjectId
- rating: Integer
- comment: String
- createdAt: Date

Methods:

- updateRating(r): void

3. Relationships and Cardinalities:-

1. **User – Address:-**
   A User *has* 0..* Addresses, whereas each Address belongs to exactly one User.

2. **User – Cart:-**
   A User *owns* exactly one Cart. Each Cart corresponds to one User.

3. **Cart – CartItem:-**
   A Cart *contains* 0..* CartItems. Each CartItem is associated with one Cart.

4. **CartItem – Product:-**
   A CartItem *refers to* one Product. A Product can be referenced by 0..* CartItems.

5. **Product – OrderItem:-**
   A Product is *included in* 0..* OrderItems. Each OrderItem references exactly one Product.

6. **Order – OrderItem:-**
   An Order *contains* 1..* OrderItems. Each OrderItem belongs to one Order.

7. **User – Order:-**
   A User *places* 0..* Orders. Each Order is placed by one User.

8. **Order – Payment:-**
   An Order may be *paid via* 0..1 Payment. Each Payment corresponds to one Order.

9. **Seller – Product:-**
   A Seller *sells* 0..* Products. A Product is sold by one Seller (based on role).

10. **Product – Category:-**
    A Product *belongs to* 0..1 Category. A Category may contain many Products.

11. **User – Review:-**
    A User *writes* 0..* Reviews. Each Review is written by exactly one User.

12. **Product – Review:-**
    A Product *has* 0..* Reviews. Each Review is associated with one Product.

4. Summary:-

The GreenCart Class Diagram provides a detailed representation of the structural design of the system. It highlights the interaction between various components such as Users, Sellers, Products, Cart, Orders, Payments, and Reviews. By clearly defining the relationships and responsibilities of each class, the diagram ensures proper understanding of the system's architecture and supports effective system development and maintenance.